Tema 5: Comparación de modelos. Métodos computacionales

Comparación de modelos

Se introduce en este tema el problema de la comparación entre diferentes modelos como alternativa a los p-valores en los contrastes de hipótesis clásicos.

Factor Bayes

En estadística bayesiana se pueden usar los factores Bayes como una alternativa a los contrastes de hipótesis clásicos.

La distribución a posteriori $P(M|D)$ de un modelo M dados los datos D está dado por el teorema de Bayes

$$P(M|D) = \frac{P(D|M)P(M)}{P(D)}$$

El término $P(D|M)$ es una verosimilitud y representa la probabilidad de que los datos sean producidos bajo la aceptación del modelo M.

Si se presenta un problema de selección de modelos en el que se debe elegir entre dos posibles modelos, en base a unos datos observados D, la plausibilidad de los dos diferentes modelos M_1 y M_2, parametrizados por vectores de parámetros θ_1 y θ_2 se puede medir mediante el factor Bayes.

El factor Bayes B se define como

$$B = \frac{P\left(D|M_1\right)}{P\left(D|M_2\right)} = \frac{\int_{\Theta_1} P\left(D|\theta_1, M_1\right) \cdot \pi\left(\theta_1|M_1\right) d\theta_1}{\int_{\Theta_2} P\left(D|\theta_2, M_2\right) \cdot \pi\left(\theta_2|M_2\right) d\theta_2}$$

donde $P\left(D|M_i\right)$ se denomina verosimilitud marginal o verosimilitud integrada.

Esto es similar a lo que se hace en los tests de la *razón de verosimilitudes* pero ahora, en lugar de de maximizar la verosimilitud, el factor Bayes realiza un *promedio* ponderado mediante la distribución de los parámetros.

Un valor de $K > 1$ significa que M_1 es apoyado por los datos más que M_2. Sin embargo, en inferencia clásica los tests de hipótesis asignan a una de las hipótesis el

status de *preferida* (hipótesis nula) y sólo se considera la evidencia que da la muestra en su contra.

El logaritmo de B recibe, a veces, el nombre de *peso de la evidencia* de D para el modelo M_1 frente a M_2.

En el caso del factor Bayes, Jeffreys estableció una escala de interpretación de B:

B	**Fuerza de la evidencia a favor de M_1**
< 1	Negativa (apoya M_2)
1 a 3	Muy escasa
3 a 10	Sustancial
10 a 30	Fuerte
30 a 100	Muy Fuerte
> 100	Decisiva

Definición Alternativa

Otra manera de considerar el concepto de factor Bayes es la siguiente:

Supongamos dos hipótesis H_0 y H_1 tales que, a priori, sus probabilidades son

$$\begin{aligned} f_0 &= P(H_0) \\ f_1 &= P(H_1) \end{aligned}$$

Después de observar una muestra, las probabilidades a posteriori de ambas hipótesis resultan ser $\alpha_0 = P(H_0|\mathbf{x})$ y $\alpha_1 = P(H_1|\mathbf{x})$.

Se define el factor Bayes a favor de H_0 como

$$B = \frac{\frac{\alpha_0}{\alpha_1}}{\frac{f_0}{f_1}} = \frac{\alpha_0 f_1}{\alpha_1 f_0}$$

Así, el factor Bayes representa la plausibilidad (*odds*) a posteriori dividida entre la plausibilidad (*odds*) a priori. Nos informa de los cambios en nuestras creencias introducidas por los datos. Tiene la propiedad de que es casi objetivo y elimina parcialmente la influencia de la distribución a priori.

Ejemplo

Supongamos el contraste simple

$H_0 \equiv \theta = \theta_0$
$H_1 \equiv \theta = \theta_1$

Tenemos que las distribuciones a posteriori son

$$\begin{aligned}\alpha_0 = P(H_0|\mathbf{x}) &= \frac{f_0 L(\theta_0|\mathbf{x})}{f_0 L(\theta_0|\mathbf{x}) + f_1 L(\theta_1|\mathbf{x})} \\ \alpha_1 = P(H_1|\mathbf{x}) &= \frac{f_1 L(\theta_1|\mathbf{x})}{f_0 L(\theta_0|\mathbf{x}) + f_1 L(\theta_1|\mathbf{x})}\end{aligned}$$

Entonces, el factor Bayes es

$$\begin{aligned}B &= \frac{\alpha_0}{\alpha_1}\frac{f_1}{f_0} \\ &= \frac{f_0 L(\theta_0|\mathbf{x})}{f_1 L(\theta_1|\mathbf{x})}\frac{f_1}{f_0} \\ &= \frac{L(\theta_0|\mathbf{x})}{L(\theta_1|\mathbf{x})}\end{aligned}$$

que coincide con la razón de verosimilitudes, de modo que la distribución a priori **no** influiría en este caso en el factor Bayes.

Ejemplo

Supongamos una v.a. que tiene como resultado *éxito* o *fracaso*. Queremos contrastar un modelo M_1 donde la probabilidad de éxito es $p = 1/2$ y otro modelo M_2 donde p es completamente desconocido de modo que tomamos una distribución a priori uniforme $U(0,1)$.

Supongamos que se toma una muestra de tamaño 200 y se obtienen 115 éxitos y 85 fallos. La verosimilitud es

$$\binom{200}{115} p^{115}(1-p)^{85}$$

De este modo,

$$\begin{aligned}P\left(X = 115|M_1\right) &= \binom{200}{115}\left(\frac{1}{2}\right)^{200} \simeq 0{,}00595 \\ P\left(X = 115|M_2\right) &= \int_0^1 \binom{200}{115} p^{115}(1-p)^{85} dp \simeq 0{,}00497\end{aligned}$$

En este caso

$$B = \frac{P\left(D|M_1\right)}{P\left(D|M_2\right)} = \frac{0{,}00595}{0{,}00497} = 1{,}197$$

de lo cual se deduce que hay una ligera evidencia a favor de M_1.

En un test de hipótesis clásico el modelo M_1 haría el papel de hipótesis nula y se tendría que rechazar a un nivel de significación $\alpha = 0{,}05$ ya que la probabilidad de obtener 115 **o más** éxitos en 200 observaciones (cuando $p = 1/2$), es igual a 0,02.

```
pbinom(114, 200, lower.tail=F, 0.5)
# sum(dbinom(115:200, 200, 0.5))   # De manera alternativa
```

```
[1] 0.0200186
```

Por tanto, si se considera un test bilateral el *p-valor* correspondiente a obtener un valor tan extremo o más que 115 es dos veces 0.02, es decir sería igual a 0.040. Por otro lado, se puede observar que 115 está a más de dos desviaciones estándar de la media que es 100.

El modelo M_2 es más complejo que M_1 porque tiene un parámetro *libre* p que le permite al modelo ajustarse mejor a los datos. La ventaja de los factores Bayes es que tienen en cuenta este hecho y eligen el modelo de máxima parsimonia o simplicidad, lo cual está más ajustado al concepto de la *navaja de Occam* (Occam razor), disminuyendo así los errores de tipo I.

Notas:

(i) Si usamos distribuciones a priori impropias para los parámetros, puede que el factor Bayes no exista.

(ii) El factor Bayes es consistente. Si H_0 es verdadera, entonces $B \to \infty$ cuando $n \to \infty$ y si H_1 es verdadera, $B \to 0$ cuando $n \to \infty$.

(iii) Hay otras alternativas: factores Bayes *fraccionales* y factores Bayes *intrínsecos*. Los dos métodos utilizan parte de los datos para crear una distribución inicial propia.

Ejemplo: Factores Bayes

Se considera la base de datos `birthwt` que tiene 189 observaciones y 10 variables. Proceden del *Baystate Medical Center* en Springfield (no donde los Simpson...).

Se trata de relacionar la variable `bwt` (birth weight in grams) con el resto de variables mediante un modelo de regresión. Para comparar los diferentes modelos se usan factores Bayes cruzados para cada uno de los modelos considerados.

```
library(MCMCpack)
data(birthwt)
help(birthwt)

model1 = MCMCregress(bwt~age+lwt+as.factor(race)+smoke+ht,
data=birthwt, b0=c(2700,0,0,-500,-500,-500,-500),
B0=c(1e-6,0.01,0.01,1.6e-5,1.6e-5,1.6e-5,1.6e-5), c0=10, d0=4500000,
marginal.likelihood="Chib95", mcmc=10000)
```

```
model2 = MCMCregress(bwt~age+lwt+as.factor(race)+smoke,
data=birthwt, b0=c(2700,0,0,-500,-500,-500),
B0=c(1e-6,0.01,0.01,1.6e-5,1.6e-5,1.6e-5),
c0=10, d0=4500000,
marginal.likelihood="Chib95", mcmc=10000)

model3 = MCMCregress(bwt~as.factor(race)+smoke+ht,
data=birthwt, b0=c(2700,-500,-500,-500,-500),
B0=c(1e-6,1.6e-5,1.6e-5,1.6e-5,1.6e-5), c0=10, d0=4500000,
marginal.likelihood="Chib95", mcmc=10000)

# Matriz de factores Bayes en la que el elemento i,j
# contiene el factor Bayes para el modelo i respecto al j
(BF = BayesFactor(model1, model2, model3))

# Probabilidades a posteriori de los modelos
PostProbMod(BF)
```

Se obtiene como resultado

```
The matrix of Bayes Factors is:
       model1 model2 model3
model1  1.000  14.08  7.288
model2  0.071   1.00  0.518
model3  0.137   1.93  1.000

The matrix of the natural log Bayes Factors is:
       model1 model2 model3
model1   0.00  2.645  1.986
model2  -2.64  0.000 -0.658
model3  -1.99  0.658  0.000

    model1     model2     model3
0.82764356 0.05879505 0.11356140
```

Bayesian Information Criterion (*BIC*)

Existen diversos autores como Kass y Raftery (1995) que sugieren no utilizar el factor Bayes y usar, en su lugar, el *BIC* que se calcula sin especificar ninguna distribución a priori.

El *BIC* se define como

$$BIC = -2 \log L\left(\hat{\theta}|\mathbf{x}\right) + p \cdot \log(n)$$

donde $L\left(\hat{\theta}|\mathbf{x}\right)$ es el valor del máximo de la función de verosimilitud, p es el número de variables explicativas del modelo (incluyendo las constantes) y n es el tamaño muestral.

Cuando se comparan dos modelos se debe **elegir** el modelo que tenga un **menor** valor de *BIC*.

Si se define la diferencia

$$S = -\frac{1}{2}\left(BIC_{M_1} - BIC_{M_2}\right) = \log L\left(\hat{\theta}_1|\mathbf{x}\right) - \log L\left(\hat{\theta}_2|\mathbf{x}\right) - \frac{1}{2}(p_1 - p_2)\log(n)$$

Si se denota como B al factor Bayes, se demuestra que

$$\frac{S - \log(B)}{\log(B)} \underset{n\to\infty}{\longrightarrow} 0$$

para tamaños muestrales grandes, de modo que se pueden aproximar los factores Bayes mediante el *BIC*.

Métodos computacionales: Introducción a MCMC

Se introducen en esta sección los métodos computacionales más empleados actualmente en estadística bayesiana: los métodos MonteCarlo de cadenas de Markov (MCMC). Estos se basan en las propiedades de las cadenas de Markov.

Nota: Un proceso estocástico discreto X_t es una cadena de Markov si $f(x_t|x_1, \ldots, x_{t-1}) = f(x_t|x_{t-1})$.

Si el espacio de estados es discreto el comportamiento de la cadena de Markov queda determinado mediante una matriz de transición, P, donde cada fila indica dónde está la cadena en cada periodo y cada columna indica a qué estado puede ir en el siguiente periodo.

Ejemplo

Supongamos que se tiene un espacio de estados bidimensional, que podría corresponder, por ejemplo, al posible voto entre dos partidos diferentes θ_1 y θ_2. Supongamos que los votantes que seleccionan θ_1 tienen una probabilidad de 0.8 de continuar haciéndolo en las siguientes elecciones, y que los que eligen θ_2 tienen una probabilidad de 0.4. Dado que hay dos posibles partidos (estados) la matriz de transición P es:

		Periodo siguiente	
		θ_1	θ_2
Periodo actual	θ_1	$\lceil 0{,}8$	$0{,}2\rceil$
	θ_2	$\lfloor 0{,}6$	$0{,}4\rfloor$

Toda cadena de Markov comienza en un valor o punto inicial; en este caso suponemos que la proporción de personas que van a elegir los partidos es igual al principio. Es decir, el punto inicial es

$$S_0 = \begin{bmatrix} 0{,}5 & 0{,}5 \end{bmatrix}$$

dado que el 50 % de la población elegiría a cada partido.

Después de la primera elección el porcentaje sería

$$S_1 = \begin{bmatrix} 0{,}5 & 0{,}5 \end{bmatrix} \begin{bmatrix} 0{,}8 & 0{,}2 \\ 0{,}6 & 0{,}4 \end{bmatrix} = \begin{bmatrix} 0{,}7 & 0{,}3 \end{bmatrix}$$

y así sucesivamente para el resto de elecciones

$$
\begin{aligned}
S_2 &= \begin{bmatrix} 0{,}7 & 0{,}3 \end{bmatrix} \begin{bmatrix} 0{,}8 & 0{,}2 \\ 0{,}6 & 0{,}4 \end{bmatrix} = \begin{bmatrix} 0{,}74 & 0{,}26 \end{bmatrix} \\
S_3 &= \begin{bmatrix} 0{,}74 & 0{,}26 \end{bmatrix} \begin{bmatrix} 0{,}8 & 0{,}2 \\ 0{,}6 & 0{,}4 \end{bmatrix} = \begin{bmatrix} 0{,}748 & 0{,}252 \end{bmatrix} \\
S_4 &= \begin{bmatrix} 0{,}748 & 0{,}252 \end{bmatrix} \begin{bmatrix} 0{,}8 & 0{,}2 \\ 0{,}6 & 0{,}4 \end{bmatrix} = \begin{bmatrix} 0{,}7496 & 0{,}2504 \end{bmatrix}
\end{aligned}
$$

Se puede observar que las proporciones convergen a $\begin{bmatrix} 0{,}75 & 0{,}25 \end{bmatrix}$. Esto es debido a que la matriz de transición tiende a un estado de equilibrio o distribución estacionaria de las proporciones. Cuando se alcanza esta situación todos los estados presentes y futuros son iguales, es decir $SP = S$, de modo que se podría obtener el resultado, en el ejemplo, simplemente como

$$
\begin{bmatrix} s_1 & s_2 \end{bmatrix} \begin{bmatrix} 0{,}8 & 0{,}2 \\ 0{,}6 & 0{,}4 \end{bmatrix} = \begin{bmatrix} s_1 & s_2 \end{bmatrix}
$$

En el caso de los métodos MCMC se trataría de ir a la *inversa*: determinar la distribución de probabilidad una vez que iterando un número suficiente de veces, se ha alcanzado la distribución estacionaria para *cualquier* punto del espacio de estados.

En general, para un espacio de estados continuo, se define la distribución estacionaria como aquella que cumple

$$
f(x) = \int f(X_t = x | X_{t-1} = y) f(y)\, dy
$$

La idea de los métodos MCMC es construir una cadena de Markov cuya distribución estacionaria sea la distribución a posteriori $f(\boldsymbol{\theta}|\mathbf{x})$ de la que se desea obtener una muestra.

Propiedades

Hay varias propiedades importantes, sobre todo respecto a definir la convergencia de los métodos de simulación. Entre ellas, se tienen:

Homogeneidad: Se dice que una cadena de Markov es homogénea en el paso m si su probabilidad de transición no depende del valor m.

Irreducibilidad: Una cadena irreducible es aquella en la que todos los estados son alcanzables desde cualquier otro estado de la cadena en un número finito de pasos. Eso implica que se puede llegar a cualquier estado desde otro estado. Es decir $P(\theta_i, \theta_j) \neq 0 \ \forall i, j$.

Recurrencia: Se dice que una cadena es recurrente cuando se puede acceder o retornar a sus estados infinitas veces. Se tiene la propiedad de que si una cadena es cerrada, finita e irreducible entonces todos los estados de la misma son recurrentes.

Estacionariedad: Si E es el conjunto de estados de una cadena de Markov con matriz de transición P, entonce se dice que una distribución es estacionaria o de equilibrio si

$$\pi = \pi P$$

Alternativamente, se puede escribir (en los casos discreto y continuo) como:

$$\begin{aligned} \pi^{t+1}(\theta_j) &= \sum_{\theta_i} P\left(\theta_i, \theta_j\right) \pi^t(\theta_i) \\ \pi^{t+1}(\theta_j) &= \int_{\Theta} P\left(\theta_i, \theta_j\right) \pi^t(\theta_i) d\theta_i \end{aligned}$$

De este modo, la distribución marginal permanece fija cuando la cadena alcanza la distribución estacionaria. Cuando se alcanza esta distribución entonces la cadena se mueve en el espacio de estados con esta distribución de manera permanente.

Ergodicidad: Se puede considerar el *periodo* de una cadena como el tiempo necesario para repetir un ciclo de valores de la cadena. Es conveniente, así, en el caso de MCMC que las cadenas sean *aperiódicas*. Si una cadena es irreducible, recurrente positiva y aperiódica entonces se dice que es ergódica. Esto equivale a decir que la cadena ya no abandona la distribución de equilibrio y que *olvida* cuál fue su primer estado inicial.

Existe un teorema de ergodicidad que es el equivalente a la ley fuerte de los grandes números pero con cadenas de Markov. Establece que cualquier función de la distribución se puede estimar mediante una muestra de la cadena de Markov obtenida a partir de un estado ergódico. Así, tras alcanzar el estado ergódico, se obtiene una muestra válida de la distribución a posteriori y, así, podemos utilizar esos datos para estimar medias, varianzas o cuantiles.

En Estadística Bayesiana el problema es el inverso: cómo construir la cadena de Markov para obtener una muestra de una distribución dada. Existen dos técnicas básicas: El muestreador de *Metropolis-Hastings* y el muestreador de *Gibbs*.

El algoritmo de Metropolis-Hastings

Este es el algoritmo más general. Se trata de simular de un parámetro del que conocemos la densidad a posteriori $f(\boldsymbol{\theta}|\mathbf{x})$ salvo, quizás, la constante de integración. Este algoritmo es semejante al algoritmo del *rechazo* para simular variables aleatorias. En este caso, se busca una densidad condicional $g(\cdot|\boldsymbol{\theta})$ o *proposal distribution* que sea fácil de muestrear. Después se generan observaciones de esta distribución de propuesta para decidir si pertenecen o no a la distribución de $\boldsymbol{\theta}|\mathbf{x}$ mediante un sorteo.

El algoritmo Metropolis-Hastings se puede resumir de la forma siguiente:

1. `Fijar un valor inicial` $\theta^{(0)}$,
2. $t = 0$,
3. `Generar` $\boldsymbol{\phi} \sim g(\cdot|\boldsymbol{\theta}^{(t)})$,
4. `Definir`
$$\alpha(\boldsymbol{\theta}^{(t)}, \boldsymbol{\phi}) = \text{mín}\left[1, \frac{f(\boldsymbol{\phi}|\mathbf{x}) \cdot g(\boldsymbol{\theta}^{(t)}|\boldsymbol{\phi})}{f(\boldsymbol{\theta}^{(t)}|\mathbf{x}) \cdot g(\boldsymbol{\phi}|\boldsymbol{\theta}^{(t)})}\right]$$
5. `Tomar`
$$\boldsymbol{\theta}^{(t+1)} = \begin{cases} \boldsymbol{\phi} & \text{con probabilidad } \alpha \\ \boldsymbol{\theta}^{(t)} & \text{en caso contrario} \end{cases}$$
6. $t = t + 1$. `Ir a 3`.

Se puede demostrar que mediante este algoritmo se simula una muestra de una cadena de Markov cuya distribución estacionaria es $f(\boldsymbol{\theta}|\mathbf{x})$.

En teoría, se puede utilizar cualquier distribución $g(\boldsymbol{\phi}|\boldsymbol{\theta})$. Lo más importante es que sea fácil de muestrear, aunque la convergencia del algoritmo será más rápida si $g \approx f$.

Comprobación de la convergencia

(i) Comparación de los resultados con valores iniciales diferentes.

(ii) Gráficos de los valores $\theta^{(t)}$.

(iii) Gráficos de la media estimada (*running means*) de θ.

(iv) Diagnósticos formales de convergencia.

Habitualmente, se usa la primera parte de los datos generados por la cadena de Markov como datos de *burn-in* para que la cadena *olvide* su estado inicial. De este modo, para la

estimación de los parámetros, sólo se utiliza la última parte de los datos generados por las cadenas.

Se estima (por ejemplo) la media a posteriori de $\boldsymbol{\theta}$ usando

$$E[\boldsymbol{\theta}|\mathbf{x}] \approx \frac{1}{N-M} \sum_{t=M+1}^{N} \boldsymbol{\theta}^{(t)},$$

donde M es el número de iteraciones de *burn-in* y N es el número total de iteraciones.

La elección de los valores iniciales no es muy importante. La convergencia puede ser más rápida si $\boldsymbol{\theta}^{(0)}$ se aproxima a la moda de la distribución final.

Para estimar la forma de la distribución final, se pueden usar estimadores *kernel* (equivalen al suavizado del histograma).

Ejemplo

Se trata de simular de una distribución de una distribución de Cauchy$(0, 1)$,

$$f(x|0,1) = \frac{1}{\pi\left(1+x^2\right)},$$

usando un algoritmo Metropolis-Hasting.

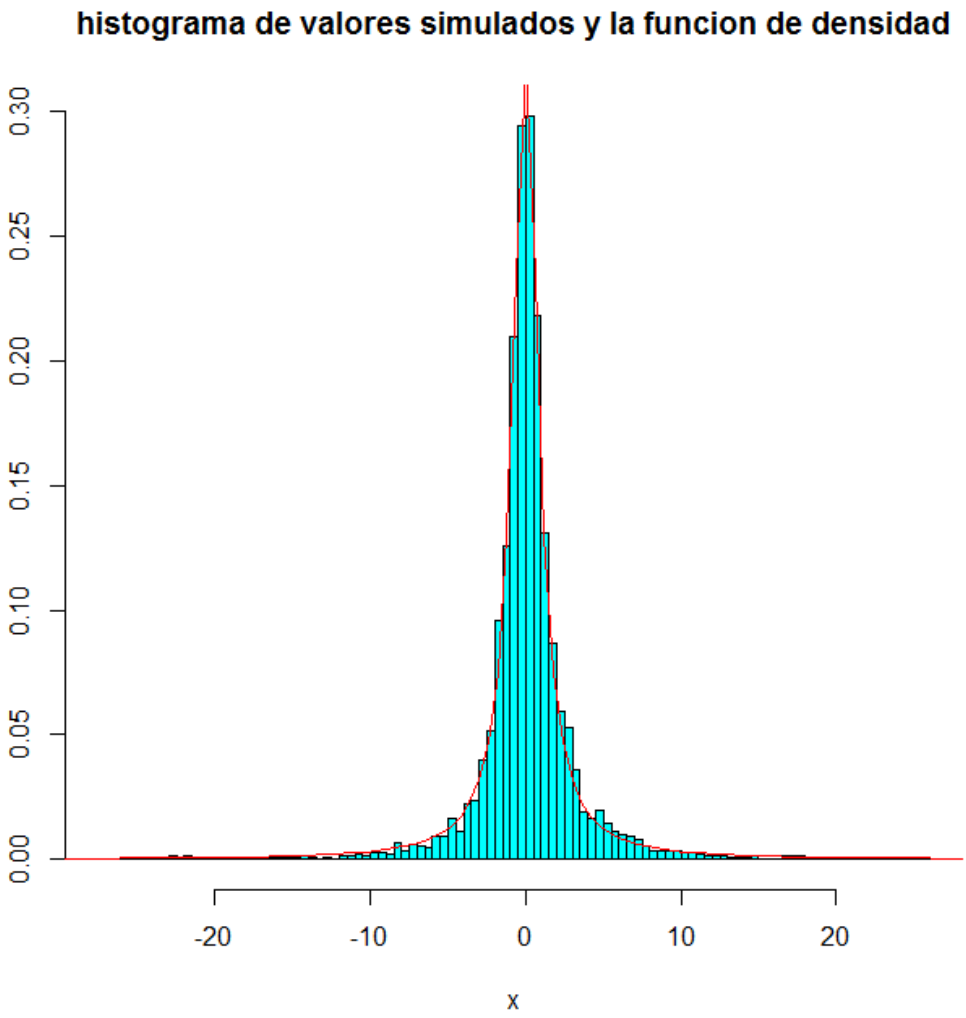
Mira en

`http://en.wikipedia.org/wiki/Cauchy_distribution`

Para la distribución propuesta consideraremos una normal con desviación estándar igual a 2:

```
sigma = 2
n = 10000
x = rep(0,n)

x[1] = rnorm(1)

for(i in 2:n){
        y = x[i-1]+sigma*rnorm(1)
        u = runif(1)

ratio =
(dcauchy(y)*dnorm(x[i-1],y,sigma))/
(dcauchy(x[i-1])*dnorm(y,x[i-1],sigma))

  alpha = min(1,ratio)
        if(u<=alpha) {x[i] = y}
        else {x[i] = x[i-1]}
        }

library(MASS)
truehist(x)
```

```
curve(dcauchy(x),-30,30,add=TRUE,col="red")
title("histograma de valores simulados y la funcion de densidad")
```



El algoritmo de Metropolis-Hasting con partición en bloques

Muchas veces puede ser difícil muestrear el vector entero de parámetros $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)$ a la vez. Sin embargo, es más fácil dividir dicho vector $\boldsymbol{\theta}$ en bloques de parámetros y muestrear por partes.

Se define,

$$\boldsymbol{\theta}_{-i}^{t} = (\theta_1^{(t+1)}, \ldots, \theta_{i-1}^{(t+1)}, \theta_{i+1}^{(t)}, \ldots, \theta_k^{(t)})$$

como el vector fijo que resulta después de que los primeros $i-1$ componentes hayan sido actualizados. Así, para simular del bloque $\boldsymbol{\theta}_i$, cuando $g_i(\cdot|\boldsymbol{\theta}_i, \boldsymbol{\theta}_{-i})$ es una distribución de propuesta, se acepta el valor $\boldsymbol{\theta}_i^{(t+1)} = \boldsymbol{\phi}$ con probabilidad

$$\alpha = \text{mín}\left[1, \frac{f(\boldsymbol{\phi}|\boldsymbol{\theta}_{-i}^{(t)}, \mathbf{x}) g_i(\boldsymbol{\theta}_i|\boldsymbol{\phi}, \boldsymbol{\theta}_{-i})}{f(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{-i}^{(t)}, \mathbf{x}) g_i(\boldsymbol{\phi}|\boldsymbol{\theta}_i, \boldsymbol{\theta}_{-i})}\right]$$

mientras que los otros componentes no cambian en esta etapa.

La distribución $f(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{-i}, \mathbf{x})$ es la distribución condicional a posteriori de $\boldsymbol{\theta}_i$.

El algoritmo tiene la ventaja que con la separación de los parámetros, se simplifica el cálculo de las probabilidades de aceptación. En muchas situaciones, es bastante fácil evaluar las distribuciones conjugadas.

El muestreador de Gibbs

El muestreador de Gibbs es una versión del método de Metropolis-Hastings donde se toma como distribución de propuesta la distribución condicionada de la distribución original de donde se quería simular el parámetro $\boldsymbol{\theta}_i$, es decir,

$$g_i(\cdot|\boldsymbol{\theta}_i, \boldsymbol{\theta}_{-i}) = f(\cdot|\boldsymbol{\theta}_{-i}, \mathbf{x})$$

Dada esta distribución, se puede demostrar que la probabilidad de aceptar $\boldsymbol{\phi}$ es siempre 1 y, así, los valores propuestos son siempre aceptados. Aunque, en este caso, las distribuciones condicionadas deben ser todas fáciles de simular.

El algoritmo para el muestreador de Gibbs es el siguiente:

1. `Comenzar con valores iniciales arbitrarios` $\boldsymbol{\theta}^{(0)}$
2. `Generar` $\theta_1^{(t+1)} \sim f(\theta_1|\mathbf{x}, \theta_2^{(t)}, \ldots, \theta_k^{(t)})$,
3. `Generar` $\theta_2^{(t+1)} \sim f(\theta_2|\mathbf{x}, \theta_1^{(t+1)}, \theta_3^{(t)}, \ldots, \theta_k^{(t)})$,

   ...

   `Generar` $\theta_k^{(t+1)} \sim f(\theta_k|\mathbf{x}, \theta_1^{(t+1)}, \ldots, \theta_{k-1}^{(t+1)})$,
4. `Ir a` 2.

Ejemplo

Se hace un estudio sobre el número de accidentes en las minas de carbón durante 112 años en Gran Bretaña. Se observa que hay un número elevado de accidentes en los primeros años y un número relativamente bajo en los últimos. La pregunta que se plantea es cuándo las mejoras en la tecnología tuvieron un efecto positivo en el número de accidentes.

Los datos obtenidos entre los años 1851–1962 son los siguientes:

4	5	4	1	0	4	3	4	0	6	3	3	4	0	2	6	3	3	5
4	5	3	1	4	4	1	5	5	3	4	2	5	2	2	3	4	2	1
3	2	2	1	1	1	1	3	0	0	1	0	1	1	0	0	3	1	0
3	2	2	0	1	1	1	0	1	0	1	0	0	0	2	1	0	0	0
1	1	0	2	3	3	1	1	2	1	1	1	1	2	4	2	0	0	0
1	4	0	0	0	1	0	0	0	0	0	1	0	0	1	0	1		

El objetivo, desde el punto de vista estadístico, es estimar el momento de cambio en la tendencia (*changepoint*) y también obtener las estimas de los parámetros de la intensidad de los desastres suponiendo dos procesos de Poisson separados (ya que son recuentos de accidentes).

De modo específico, tenemos una serie de recuentos $x_1, x_2, \ldots, x_n$ donde aparece una posibilidad de cambio en un momento dado k. Se tienen, así, dos procesos de Poisson:

$$\begin{cases} x_i|\lambda \sim \text{Po}(\lambda) & \text{para } i = 1, \ldots, k \\ x_i|\phi \sim \text{Po}(\phi) & \text{para } i = k+1, \ldots, n \end{cases}$$

donde la decisión de qué modelo aplicar depende de cuál sea el punto k.

Se tienen que estimar tres parámetros: λ, ϕ y k. En principio, las distribuciones a priori que se consideran son

$$\begin{aligned} \lambda &\sim \text{Ga}(\alpha, \beta) \\ \phi &\sim \text{Ga}(\gamma, \delta) \\ k &\sim \text{UD}[1, 2, \ldots, n] \end{aligned}$$

donde UD es una distribución uniforme discreta. Se toman como valores de los parámetros de las distribuciones a priori:

$$\begin{aligned} \alpha &= 4 \\ \beta &= 1 \\ \gamma &= 1 \\ \delta &= 2 \end{aligned}$$

La distribución a posteriori es

$$\begin{aligned}
&\pi\left(\lambda,\phi,k|\mathbf{y}\right) \propto L\left(\lambda,\phi,k|\mathbf{y}\right)\cdot\pi\left(\lambda|\alpha,\beta\right)\cdot\pi\left(\phi|\gamma,\delta\right)\cdot\pi(k)\propto\\
\propto\ & \left(\prod_{i=1}^{k}\frac{e^{-\lambda}\lambda^{y_i}}{y_i!}\right)\cdot\left(\prod_{i=k+1}^{n}\frac{e^{-\phi}\phi^{y_i}}{y_i!}\right)\cdot\left(\frac{\beta^{\alpha}}{\Gamma(\alpha)}\lambda^{\alpha-1}e^{-\beta\lambda}\right)\cdot\left(\frac{\delta^{\gamma}}{\Gamma(\gamma)}\phi^{\gamma-1}e^{-\delta\phi}\right)\cdot\frac{1}{n}\\
\propto\ & \exp\left\{-\lambda k\right\}\cdot\left(\frac{\lambda^{\sum_{i=1}^{k}y_i}}{\prod_{i=1}^{k}y_i!}\right)\cdot\exp\left\{-\phi(n-k)\right\}\cdot\left(\frac{\phi^{\sum_{i=k+1}^{n}y_i}}{\prod_{i=k+1}^{n}y_i!}\right)\cdot\lambda^{\alpha-1}e^{-\beta\lambda}\cdot\phi^{\gamma-1}e^{-\delta\phi}\\
\propto\ & \lambda^{\alpha-1+\sum_{i=1}^{k}y_i}\cdot\phi^{\gamma-1+\sum_{i=k+1}^{n}y_i}\cdot\exp\left\{-k(\lambda-\phi)-\lambda\beta-\phi(\delta+n)\right\}.
\end{aligned}$$

A partir de la expresión anterior se obtienen de manera directa las distribuciones condicionadas:

$$\begin{aligned}
\pi\left(\lambda|\phi,k,\mathbf{y}\right) &\propto \lambda^{\alpha+\sum_{i=1}^{k}y_i-1}\cdot\exp\left\{-\lambda(k+\beta)\right\}\Longrightarrow\\
\lambda|\phi,k &\sim Ga\left(\alpha+\sum_{i=1}^{k}y_i,\ \beta+k\right)
\end{aligned}$$

$$\begin{aligned}
\pi\left(\phi|\lambda,k,\mathbf{y}\right) &\propto \phi^{\gamma+\sum_{i=k+1}^{n}y_i-1}\cdot\exp\left\{-\phi(\delta+n-k)\right\}\Longrightarrow\\
\phi|\lambda,k &\sim Ga\left(\gamma+\sum_{i=k+1}^{n}y_i,\ \delta+n-k\right)
\end{aligned}$$

La distribución condicionada de k resulta ser la más complicada. Si denominas $\sum_{i=1}^{n}y_i = M$, entonces

$$\sum_{i=1}^{k}y_i+\sum_{i=k+1}^{n}y_i=M\Rightarrow\sum_{i=k+1}^{n}y_i=M-\sum_{i=1}^{k}y_i$$

de modo que

$$\lambda^{\sum_{i=1}^{k}y_i}\cdot\phi^{\sum_{i=k+1}^{n}y_i}=\lambda^{\sum_{i=1}^{k}y_i}\cdot\phi^{M-\sum_{i=1}^{k}y_i}=\left(\frac{\lambda}{\phi}\right)^{\sum_{i=1}^{k}y_i}\phi^{M}$$

donde ϕ^M actua como una cosntante en la distribución condicionada de k. Así, la distribución de k resulta después de simplificar,

$$\pi\left(k|\lambda,\phi\right)\propto\exp\left\{-k(\lambda-\phi)\right\}\left(\frac{\lambda}{\phi}\right)^{\sum_{i=1}^{k}y_i}.$$

El programa en `R` que se puede utilizar es el siguiente:

```
desastres = c(4,5,4,1,0,4,3,4,0,6,3,3,4,0,2,6,3,3,
              5,4,5,3,1,4,4,1,5,5,3,4,2,5,2,2,3,4,2,1,
              3,2,2,1,1,1,1,3,0,0,1,0,1,1,0,0,3,1,0,
              3,2,2, 0,1,1,1,0,1,0,1,0,0,0,2,1,0,0,0,
              1,1,0,2,3,3,1,1,2,1,1,1,1,2,4,2,0,0,0,1,
              4,0,0,0,1,0,0,0,0,0,1,0,0,1,0,1)
```

```
# Generador del parametro k
k.post.dist = function(t1, t2, X){
n = length(X)
post.dist = rep(0,n)
for (i in 1:n) {
post.dist[i] = exp(-(t1-t2)*i)*(t1/t2)^sum(X[1:i])}
post.dist = post.dist/sum(post.dist)
sal = sample(1:n,1,prob=post.dist)
return(sal)
}


# Muestrador de Gibbs
gibbsD = function(teta,y,reps)  {
alfa = 4; beta = 1; gamma = 1; delta = 2
for (i in 2:(reps+1))  {
   lambda = rgamma(1,alfa+sum(y[1:teta[(i-1),3]]),
          (beta+teta[(i-1),3]))
   fi = rgamma(1,gamma+sum(y[teta[(i-1),3]:length(y)]),
         (delta+length(y)-teta[(i-1),3]))
    m = k.post.dist(lambda,fi,y)
        teta = rbind(teta,c(lambda,fi,m))
  }
        return(teta)
}
# Se ejecuta el algoritmo del Gibbs
empiezo = matrix(c(1,1,100),1,3)
salida = gibbsD(empiezo,desastres,10000)
salida[,3] =  salida[,3] + 1851
lista = summary(salida[2001:10001,])
colnames(lista) = c("     lambda","     fi","     anno")
lista
```

Se obtienen los siguientes valores:

```
     lambda          fi              anno
Min.   :2.054   Min.   :0.5019   Min.   :1881
1st Qu.:2.943   1st Qu.:0.8627   1st Qu.:1889
Median :3.135   Median :0.9387   Median :1891
Mean   :3.145   Mean   :0.9437   Mean   :1891
3rd Qu.:3.331   3rd Qu.:1.0186   3rd Qu.:1892
Max.   :4.306   Max.   :1.4543   Max.   :1900
```

Alternativamente se puede usar el programa (recogido en la web de libro *Bayesian Statistics* de Lee):

`http://www-users.york.ac.uk/~pml1/bayes/rprogs/coal.txt`

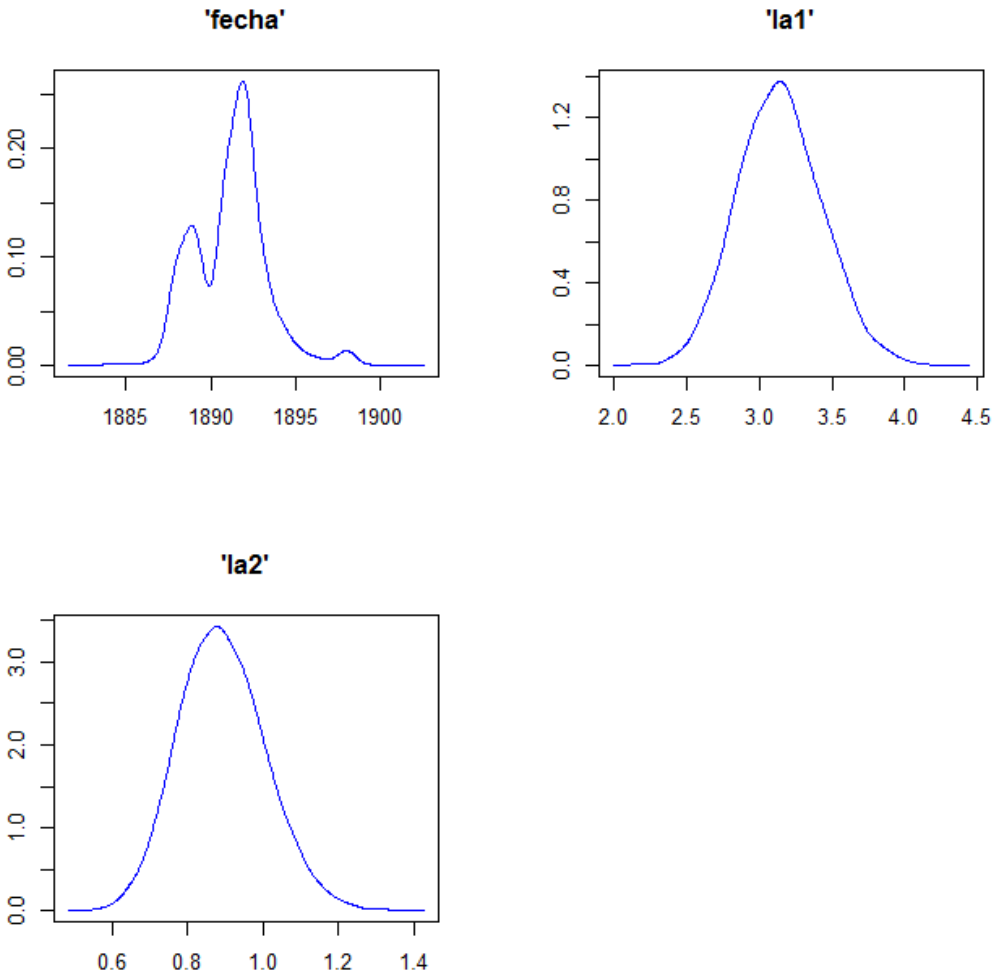
Modelo con OpenBugs

El programa que utiliza `R2Openbugs` o `BRugs` correspondiente es:

```
rm(list=ls(all=TRUE))
ruta = "C:/Directorio_Trabajo"
setwd(ruta)

#.....................................................................
# PROGRAMA de BUGS
#.....................................................................
cat("
model{
for(ano in 1:N){
D[ano] ~ dpois(mu[ano])
log(mu[ano]) <- b[1] + step(ano-cambiaano)*b[2]
}

for (j in 1:2){b[j] ~ dnorm(0,1.0E-6)}
cambiaano ~ dunif(1,N)

fecha <- 1851.5 + cambiaano
la1  <- exp(b[1])
la2 <- exp(b[1] + b[2])
}
",
file="modelo.txt")
#.....................................................................

D=c(4,5,4,1,0,4,3,4,0,6,3,3,4,0,2,6,3,3,5,4,5,3,1,4,4,1,5,5,3,4,
2,5,2,2,3,4,2,1,3,2,1,1,1,1,1,3,0,0,1,0,1,1,0,0,3,1,0,3,2,2,0,1,
1,1,0,1,0,1,0,0,0,2,1,0,0,0,1,1,0,2,2,3,1,1,2,1,1,1,1,2,4,2,0,0,
0,1,4,0,0,0,1,0,0,0,0,0,1,0,0,1,0,0)

N=length(D)
datos=list("N","D")

iniciales = function(){list(b=rnorm(2,0,1),cambiaano=rpois(1,20))}
parametros = c("fecha","la1","la2")

# Con BRugs
library(BRugs)
minas = BRugsFit(data=datos, inits=iniciales, para=parametros,
nBurnin=10000, nIter=10000, modelFile="modelo.txt",
numChains=1, working.directory=ruta)

samplesStats("*")          # Resultados finales
samplesDensity("*", mfrow=c(2,2), col=4)
```

```
# Con OpenBugs
library(R2OpenBUGS)
modelo = bugs(data=datos,inits=iniciales,
parameters.to.save=parametros,
model.file="modelo.txt",
n.chains=1, n.iter=20000, n.burnin=10000,
working.directory=ruta, clearWD=TRUE, debug=TRUE)

print(modelo)
```

```
          mean     sd MC_error val2.5pc    median val97.5pc start sample
fecha 1891.0000 2.1800 0.028480 1888.000 1891.0000  1896.000 10001  10000
la1      3.1430 0.2926 0.004308    2.589    3.1360     3.732 10001  10000
la2      0.8925 0.1142 0.001400    0.685    0.8866     1.129 10001  10000
```



El programa que utiliza `MCMCpack`

```r
# Datos
Y = c(4,5,4,1,0,4,3,4,0,6,3,3,4,0,2,6,3,
3,5,4,5,3,1,4,4,1,5,5,3,4,2,5,2,2,3,4,2,1,
3,2,1,1,1,1,1,3,0,0,1,0,1,1,0,0,3,1,0,3,2,
2,0,1,1,1,0,1,0,1,0,0,0,2,1,0,0,0,1,1,0,2,
2,3,1,1,2,1,1,1,1,2,4,2,0,0,0,1,4,0,0,0,1,
0,0,0,0,0,1,0,0,1,0,0)

library(MCMCpack)

m = MCMCpoissonChange(Y ~ 1, m=1, c0=1, d0=1,
marginal.likelihood="Chib95")

summary(m)

X11()
plot(m)

X11()
plotChangepoint(m, verbose=TRUE, ylab="Density", start=1,
overlay=TRUE)

X11()
plotState(m)
```

```
Iterations = 1001:2000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

                     Mean     SD Naive SE Time-series SE
(Intercept)_regime1 3.0827 0.2881 0.009111       0.009638
(Intercept)_regime2 0.8904 0.1125 0.003556       0.003556

2. Quantiles for each variable:

                     2.5%    25%    50%    75% 97.5%
(Intercept)_regime1 2.5516 2.8696 3.0759 3.2540 3.706
(Intercept)_regime2 0.6797 0.8134 0.8839 0.9644 1.119
```

Ejemplo de Factores Bayes y problemas tipo *changepoint*

```
library(MCMCpack)

# Datos
Y=c(4,5,4,1,0,4,3,4,0,6,3,3,4,0,2,6,3,3,5,4,5,3,1,4,4,
    1,5,5,3,4,2,5,2,2,3,4,2,1,3,2,1,1,1,1,1,3,0,0,1,0,
    1,1,0,0,3,1,0,3,2,2,0,1,1,1,0,1,0,1,0,0,0,2,1,0,0,
    0,1,1,0,2,2,3,1,1,2,1,1,1,1,2,4,2,0,0,0,1,4,0,0,0,
    1,0,0,0,0,0,1,0,0,1,0,0)

model1 = MCMCpoissonChange(Y ~ 1, m=1, c0=6.85, d0=1, burnin=5000,
mcmc=5000, verbose=1000, marginal.likelihood="Chib95")

model2 = MCMCpoissonChange(Y ~ 1, m=2, c0=6.85, d0=1, burnin=5000,
mcmc=5000, verbose=1000, marginal.likelihood="Chib95")

model3 = MCMCpoissonChange(Y ~ 1, m=3, c0=6.85, d0=1, burnin=5000,
mcmc=5000, verbose=1000, marginal.likelihood="Chib95")

print(BayesFactor(model1, model2, model3))

# Grafica del modelo mejor ajustado
plotState(model1)

plotChangepoint(model1, verbose=TRUE, ylab="Density", start=1,
overlay=TRUE)
```

```
The matrix of Bayes Factors is:
        model1 model2 model3
model1 1.00000 7.9266  126.4
model2 0.12616 1.0000   15.9
model3 0.00791 0.0627    1.0

The matrix of the natural log Bayes Factors is:
       model1 model2 model3
model1   0.00   2.07   4.84
model2  -2.07   0.00   2.77
model3  -4.84  -2.77   0.00

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Expected changepoint(s)  40
Local means for each regime are  3.153846 0.890411
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
```

Modelos Jerárquicos

Los datos que presentan algún tipo de jerarquía son habituales en muchos contextos donde aparecen diferentes niveles de agregación. Por ejemplo, en datos de tipo sociológico se pueden tener en cuenta variables medidas en diferentes zonas geográficas. Los modelos jerárquicos identifican grupos naturales espaciales o temporales modelizando así dependencias entre las observaciones.

La forma de usar modelos jerárquicos es suponer que los datos dependen de una serie de parámetros que, a su vez, dependen de otros parámetros con sus respectivas funciones de probabilidad.

De modo teórico, se supone que el vector de parámetros $\boldsymbol{\theta}$ tiene una distribución condicionada por otro vector de parámetros $\boldsymbol{\psi}$, de modo que la distribución a posteriori es

$$\pi\left(\boldsymbol{\theta}, \boldsymbol{\psi}|\mathbf{x}\right) \propto L\left(\boldsymbol{\theta}|\mathbf{x}\right) \cdot P\left(\boldsymbol{\theta}|\boldsymbol{\psi}\right) \cdot P(\boldsymbol{\psi})$$

En este caso, $\boldsymbol{\theta}$ tiene como distribución a priori $P\left(\boldsymbol{\theta}|\boldsymbol{\psi}\right)$ pero, a su vez, está condicionada por otro parámetro que tiene su propia distribución a priori $P(\boldsymbol{\psi})$ llamada *hiperpriori* que tiene sus propios *hiperparámetros*.

La inferencia sobre cada parámetro se puede hacer calculando sus respectivas densidades marginales

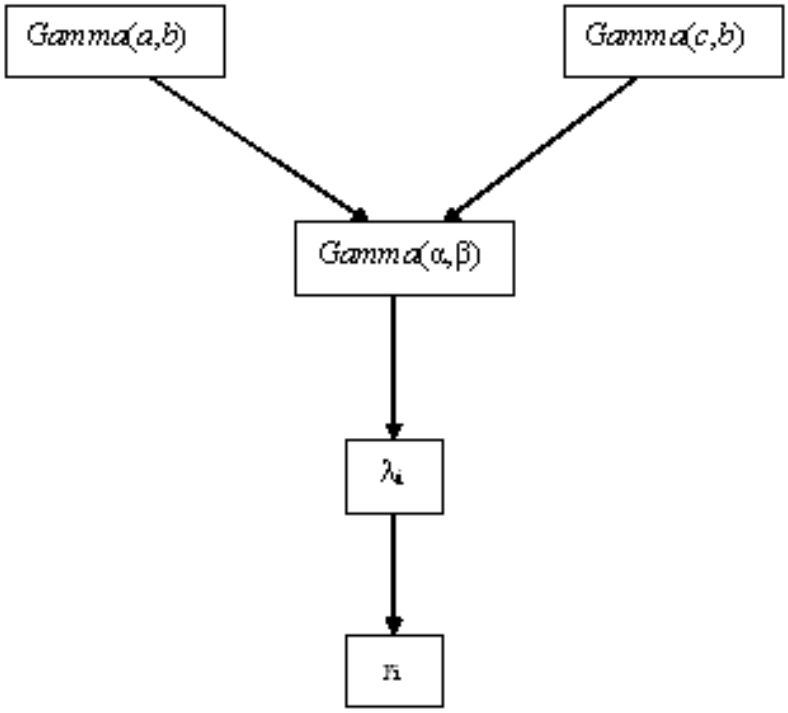
$$\begin{aligned} \pi\left(\boldsymbol{\theta}|\mathbf{x}\right) &= \int_{\Psi} \pi\left(\boldsymbol{\theta}, \boldsymbol{\psi}|\mathbf{x}\right) d\boldsymbol{\psi} \\ \pi\left(\boldsymbol{\psi}|\mathbf{x}\right) &= \int_{\Theta} \pi\left(\boldsymbol{\theta}, \boldsymbol{\psi}|\mathbf{x}\right) d\boldsymbol{\theta} \end{aligned}$$

Habitualmente, este tipo de integrales son difíciles de hacer analíticamente por lo que es necesario aplicar métodos MCMC.

Se podría considerar un número ilimitado de niveles de jerarquía, aunque se acepta que no suelen existir razones suficientes para que se supongan más de dos niveles.

Modelo jerárquico Poisson–gamma

Supongamos que los datos observados proceden de una distribución de Poisson. La distribución a priori para el parámetro λ se asume que es una gamma para obtener un modelo conjugado. A su vez, esta gamma tiene dos parámetros α y β a los que se puede asignar también una distribución de probabilidad para reflejar diferencias en el segundo nivel de jerarquía. Una posible opción sería considerar otras distribuciones gamma para estos parámetros con sus propios parámetros fijados (hiperparámetros).



En este caso, el parámetro de intensidad de la Poisson tiene un índice i, es decir, se escribe como λ_i porque ahora no se asume que es un efecto fijo que que tiene un impacto constante sobre los datos sino que se supone que λ varía en relación los datos: $\lambda_i \sim Gamma(\alpha, \beta)$.

De este modo, en lugar de muestrear de un proceso que se modeliza mediante una función de densidad, ahora cada observación procede de una serie de distribuciones relacionadas mediante una distribución común:

$$\begin{aligned} y_i &\sim Po\left(\lambda_i\right) \\ \lambda_i &\sim Gamma(\alpha, \beta) \\ \alpha &\sim Gamma(a, b) \\ \beta &\sim Gamma(c, d) \end{aligned}$$

donde α y β son independientes entre sí.

En este caso,

$$\begin{aligned} P\left(\mathbf{y}, \lambda_i, \alpha, \beta\right) &= \prod_{i=1}^{n} P\left(y_i|\lambda_i\right) \cdot P\left(\lambda_i|\alpha, \beta\right) \cdot P\left(\alpha|a, b\right) \cdot P\left(\beta|c, d\right) \propto \\ &\propto \lambda_i^{\sum_i y_i+\alpha-1} \alpha^{a-1}\beta^{\alpha+c-1}\Gamma(\alpha)^{-1} \exp\left[-\lambda_i(1+\beta) - \alpha b - \beta d\right] \end{aligned}$$

lo cual no es manejable desde el punto de vista analítico y se necesita usar un método MCMC.

Ejemplo

Se consideran los datos del número de matrimonios cada 1000 personas en Italia desde el año 1936 hasta 1951. La pregunta que se plantea es si es conveniente modelizar las tasas de matrimonio durante la guerra mundial del mismo modo, antes y después de la guerra.

Los valores y_i son

7	9	8	7	7	6	6	5
5	7	9	10	8	8	8	7

En este caso no parece lógico considerar que la intensidad de la distribución de Poisson es constante durante todos esos años. En su lugar, se supone que λ_i son diferentes en cada año, aunque proceden de una distribución común.

Se asume distribuciones a priori $Gamma(1, 1)$ para los parámetros, y se considera el siguiente modelo en `OpenBugs` o `BRugs`:

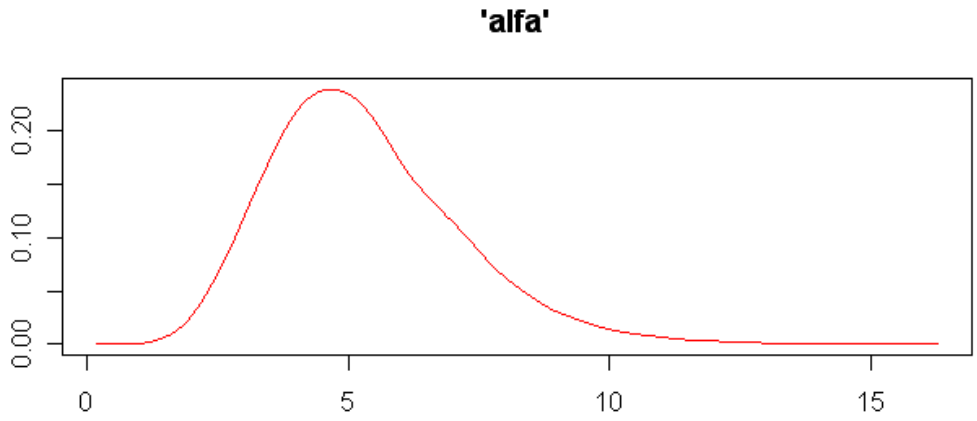
```
rm(list=ls(all=TRUE))
ruta = "C:/Directorio_Trabajo"
setwd(ruta)

#.................................................................
# PROGRAMA de BUGS
#.................................................................
cat("
model italia
    {
    for (i in 1:n) {
        lambda[i] ~ dgamma(alfa,beta)
        y[i] ~ dpois(lambda[i])
    }

    # Priors
    alfa ~ dgamma(1,1)
    beta ~ dgamma(1,1)
}
",
file="modelo.txt")
#.................................................................
```

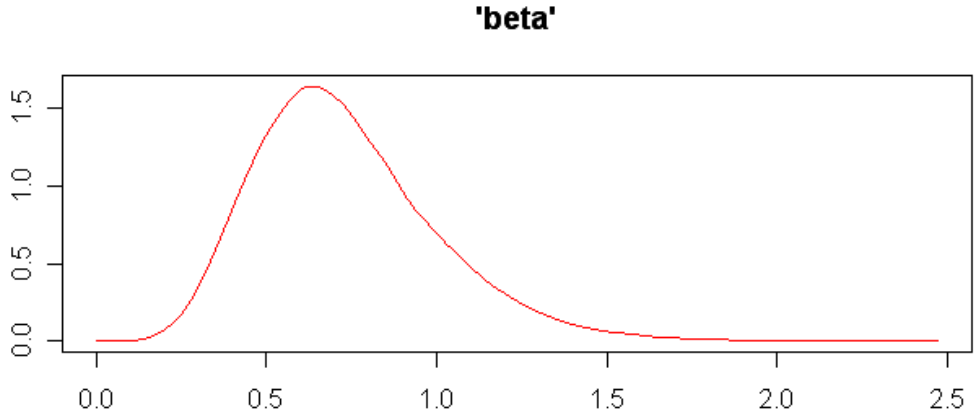
```
# Datos
y = c(7,9,8,7,7,6,6,5,5,7,9,10,8,8,8,7)
n = length(y)
datos = list("y","n")

parametros = c("alfa","beta","lambda")

iniciales = function(){list(alfa=rgamma(1,1,.1),
beta=rgamma(1,1,.1), lambda=runif(n))}
# modelGenInits()

nburnin = 10000
ntotaliter = 10000

library(BRugs)

# Con BRugs
modelo = BRugsFit(data=datos, inits=iniciales, para=parametros,
nBurnin=nburnin, nIter=ntotaliter, modelFile="modelo.txt",
numChains=3, working.directory=ruta)

samplesStats("*")

X11()
par(mfrow=c(2,1))
plotDensity("alfa",xlab=expression(alpha), main="Densidad de alfa")
plotDensity("beta",xlab=expression(beta), main="Densidad de beta")

X11()
samplesDensity("lambda", beg=(nburnin+1), end=samplesGetEnd(),
mfrow=c(4,4), ask=FALSE)

# Con OpenBugs
library(R2OpenBUGS)

modelo = bugs(data=datos,inits=iniciales,
parameters.to.save=parametros,
model.file="modelo.txt",
n.chains=1, n.iter=20000, n.burnin=10000,
working.directory=ruta, clearWD=TRUE, debug=TRUE)

print(modelo)
```

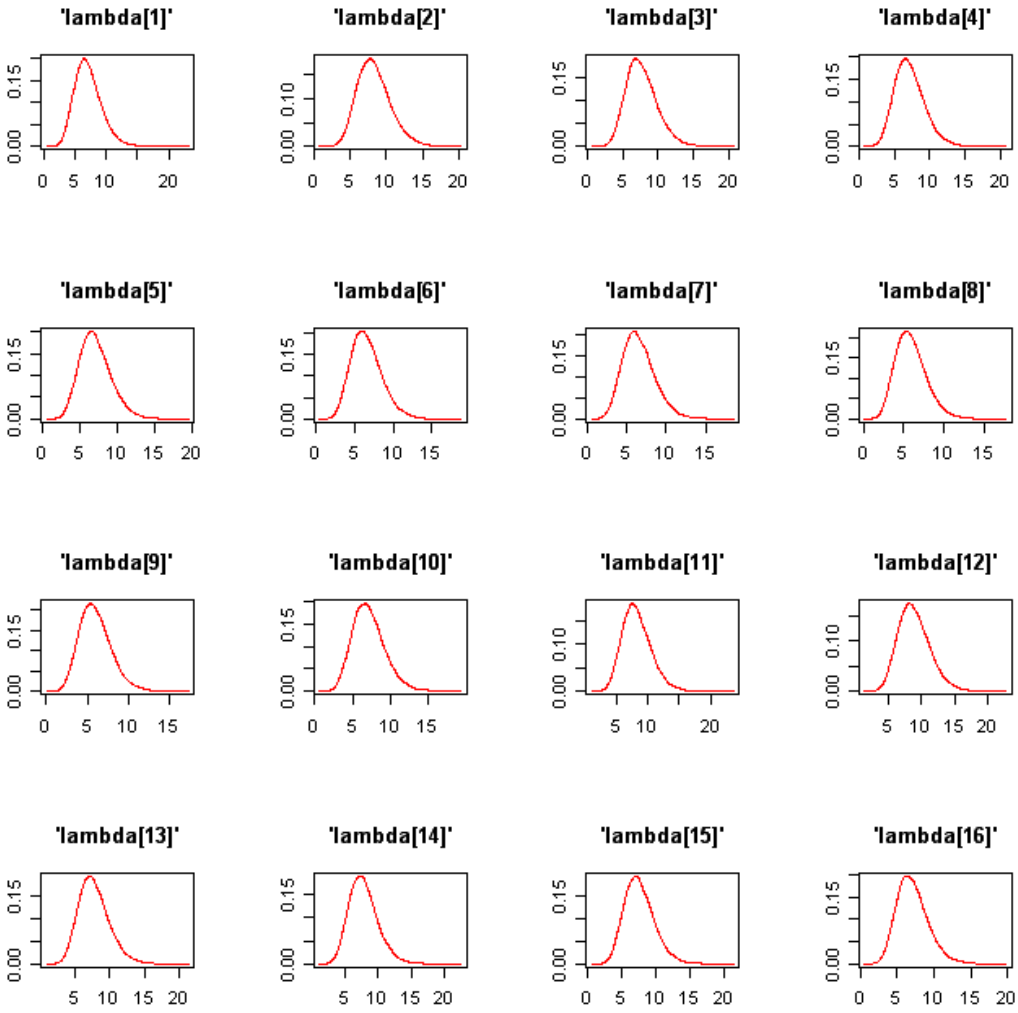
Se obtiene como resultado

	mean	sd	MC_error	val2.5pc	median	val97.5pc	start	sample
alfa	5.4340	1.9360	0.050320	2.4000	5.1820	9.798	10001	30000
beta	0.7539	0.2873	0.007456	0.3108	0.7152	1.402	10001	30000
lambda[1]	7.1280	2.0830	0.013970	3.6480	6.9270	11.760	10001	30000
lambda[2]	8.2660	2.2580	0.015060	4.4740	8.0530	13.300	10001	30000
lambda[3]	7.6950	2.1770	0.014090	4.0910	7.4920	12.520	10001	30000
lambda[4]	7.1100	2.0690	0.013120	3.6580	6.8960	11.700	10001	30000
lambda[5]	7.1060	2.0860	0.012660	3.6580	6.8840	11.760	10001	30000
lambda[6]	6.5160	1.9820	0.011610	3.2010	6.3280	10.950	10001	30000
lambda[7]	6.5370	1.9870	0.012610	3.2420	6.3500	10.940	10001	30000
lambda[8]	5.9330	1.8940	0.012820	2.7900	5.7460	10.140	10001	30000
lambda[9]	5.9510	1.9190	0.012910	2.7920	5.7600	10.260	10001	30000
lambda[10]	7.1100	2.0880	0.012980	3.6010	6.9110	11.700	10001	30000
lambda[11]	8.3090	2.2950	0.014830	4.4890	8.0760	13.390	10001	30000
lambda[12]	8.8730	2.3540	0.016020	4.9170	8.6480	14.120	10001	30000
lambda[13]	7.6750	2.1680	0.014240	4.0380	7.4740	12.480	10001	30000
lambda[14]	7.6770	2.1590	0.013320	4.0390	7.4730	12.470	10001	30000
lambda[15]	7.6880	2.1620	0.013350	4.0850	7.4780	12.490	10001	30000
lambda[16]	7.0910	2.0800	0.012220	3.6030	6.8800	11.740	10001	30000

Y las gráficas de las funciones de densidad estimadas son







Se observa que los parámetros tienen una distribución unimodal y casi simétrica. En el caso de los valores de λ_i $(i = 1936, \cdots, 1951)$ se observa una bajada de las tasas de matrimonios destacable durante la segunda guerra mundial.

Ejemplo simulado de varios grupos

Se supone un serie de grupos (por ejemplo diferentes barrios) y una serie de medidas tomadas entre una muestra de sus habitantes.

```
rm(list=ls(all=TRUE))        # Limpio la memoria
ruta = "c:/dondesea"
setwd(ruta)                  # Fijo la ruta de trabajo

# Se generan unos datos artificiales
Nj = c(41, 37, 42, 40)
Ntot = sum(Nj)
muJ = rep(c(-1, 0, 1, 2), Nj)
MisDatos = data.frame(grupo=rep(1:4, Nj),
datos = rnorm(Ntot, muJ, 2))

y = MisDatos[,2]
grupo = MisDatos[,1]
n = length(y)

#.....................................................

modelo = function(){
# Modelo normal jerarquico

for(i in 1:n) {
        y[i] ~ dnorm(teta[grupo[i]], tao)
}

for(j in 1:J) {
        teta[j] ~ dnorm(mu, tao.b)
}

mu ~ dnorm(0, 0.0001)
tao ~ dunif(0, 100)
tao.b ~ dunif(0, 100)
sigma2 <- 1/tao
sigma2.b <- 1/tao.b
}

#.............................................

datos = list("y", "grupo", "n", "J")
parametros = c("mu", "sigma2", "sigma2.b", "teta")
```

```
library(BRugs)

sale = BRugsFit(data=datos, inits=NULL, para=parametros,
nBurnin = 10000, nIter = 10000, modelFile=modelo,
numChains=3,working.directory=ruta)

samplesStats("*") # Resultados finales
samplesDensity("*", mfrow=c(3,3), col=4)
```

```
            mean     sd MC_error val2.5pc  median val97.5pc
mu        0.6882 0.6156 0.003864  -0.5433  0.6854    1.9220
sigma2    4.7360 0.5398 0.003393   3.8020  4.6950    5.9040
sigma2.b  1.3380 1.7140 0.013890   0.2109  0.8913    5.2750
teta[1]  -0.5545 0.3492 0.002472  -1.2330 -0.5569    0.1363
teta[2]   0.2117 0.3417 0.002155  -0.4595  0.2136    0.8779
teta[3]   1.0840 0.3238 0.001844   0.4501  1.0810    1.7220
teta[4]   1.9960 0.3540 0.002597   1.2940  1.9990    2.6840
```