BAY - 04) JAGS - Introducción al muestreo GIBS y el método MCMC

Eduardo García Tapia

5.1) Introducción

El método *Monte Carlo - Markov Chain* es un conjunto de métodos estadísticos, probabilísticos y de simulación que se utilizan para muestrear de una función de distribución de probabilidad conjunta $p(\boldsymbol{\theta}) = p(\theta_1, \theta_2, ..., \theta_k)$; se utiliza cuando la dimensión del espacio de probabilidad es alto y la función compleja. Muestrear a partir de distribuciones de probabilidad conjunta no es una tarea trivial; incluso puede complicarse en el caso más sencillo como una normal multivariada $N_2(\boldsymbol{\mu}, \Sigma)$.

En un contexto bayesiano, la posibilidad de utilizar algoritmos como el MCMC para muestrear de la distribución final *(a posterior)* con la forma $p(\theta|x_1, ..., x_n)$, es la que habilita el análisis para distribuciones que no son conjugadas o conocidas. Por eso, los métodos de simulación son la piedra angular en la inferencia bayesiana.

Suponiendo que se tiene un vector inicial para $p(\boldsymbol{\theta})$ con valores iniciales $({\theta_1}^{(step\ 0}, {\theta_2}^{step\ 0}, ..., \theta^0)$, después, se propone un punto nuevo para θ_1, utilizando Monte Carlo, a partir de la función $p({\theta_1}^{step\ 1}|{\theta_2}^0, ..., {\theta_k}^0)$. Se calcula una razón de aceptación, la cuál establece, bajo cierto nivel de probabilidad si la simulación es válida, y si debe conservarse; es decir, que si la simulación pertenece a la distribución o no. Depués se simula un punto para ${\theta_2}^1$ con la distribución $p({\theta_2}^1|{\theta_1}^1, {\theta_3}^0, ..., {\theta_k}^0)$, y así sucesivamente.

El nombre de *Markov Chain* surge a partir de que cada paso siguiente solo depende del anterior. Entonces, el algoritmo se basa en ir muestreando a partir de condicionales (distirbuciones más sencillas) iterativamente hasta intentar alcanzar un estado de estacionariedad en la cadena de Markov, en el que se supone que siempre se estaría muestreando de la distribución objetivo.

La implementaación de inferencia bajo el enfoque bayesiano utiliza las técnicas como MCMC y Gibbs Sampling. Para los ejercicios que se realizan en esta serie de notas se utiliza BUGS *(Bayesian inference Using Gibbs Sampling)*, una lenguaje probabilístico que permite utilizar JAGS *(Just another Gibbs Sampler)*, una librería en R que realiza los cálculos necesarios.

5.2) El modelo binomial

Contexto: Se hace inferencia sobre una distribución binomial, utilizando el enfoque bayesiano. La distribuciones para el proceso de inferencia son los siguientes. Recuerde que los hiperparámetros a y b sirven para definir una distribución inicial informativa o no. Note que la distribución Binomial es, en realidad, la suma de un conjunto de distribuciones Bernoulli.

Sea $X \sim Ber(\theta)$ *y* $Y \sim Bin(\theta, n) = \sum_{i=1}^{n} X_i$

$$Muestral: \quad p(y|\theta)\ \alpha\ \theta^x(1-\theta)^{n-x}\ \alpha$$
$$Inicial: \quad p(\theta) \sim Beta(a, b)\ \alpha\ \theta^a(1-\theta)^b$$

Recuerde que el objetivo es obtener muestras de $p(\theta|x)$ *a partir de las cuales se pueda conocer su distribución.*

Paso 1. Definir los datos muestrales

Los datos muestrales son definidos y asignados a una variable en R, a la vez que se almacena el nombre de las variables que los contienen.

```
# Librerías necesarias
library(foreign)
library(lattice)
library(R2jags)

# Semilla para generar números aleatorios
```

```r
set.seed(1234)

# Se definen los datos observados, a partir de la distribución Bernoulli que los genera
datos = list("n","x")
n = 20
x = c(1,1,1,0,1,0,1,1,1,0,0,1,1,1,0,1,1,1,1,1)
```

Paso 2. Definir el modelo probabilístico.

Guarda la especificacion modelo en el directorio de trabajo como un archivo de texto con terminación *".jags"*. Se define la distribución de los datos muestrales y la inicial del parámetro de interés. La librería JAGS se encarga de muestrear de las distribuciones de las cantidades de interés. Éstas pueden ser parámetros, transformaciones de éstos, valores a predecir, etc. Note que las distribuciones se almacenan solo en el archvio de texto y no directamente en variables de R.

```r
# Define las cantidades de interés para muestrear
cantidades_interes = c("theta","phi","y_predecida")

# Iniciar creación del archivo ".jags" donde se guarda el modelo
cat("model {

# Definir cada respuesta con distribución Bernoulli
for(i in 1:n)
    {x[i] ~ dbern(theta)}

# Distribución inicial
theta ~ dbeta(9.2, 13.8)

# Logaritmo de los momios
phi = log(theta / (1 - theta))

# Distribucion predictiva
y_predecida ~ dbin(theta, 40)

# Cerrar creación del archivo
}", fill=TRUE, file="BAY - 05.jags")
```

Paso 3. Correr el muestreo Gibbs.

Se definen los valores iniciales a partir de los cuales se arranca el método MCMC. Se definen dos valores iniciales, porque se corren dos cadenas independientes. Esto es para poder realizar un análisis de convergencia que permita conocer si el método ha logrado la estacionariedad y muestrear efectivamente de las distribuciones de interés.

```r
# Para las dos cadenas
inits_1 = list("theta"=0.25)
inits_2 = list("theta"=0.75)
iniciales_binomial = list(inits_1, inits_2)

# Se utiliza para no mostrar salida de texto en el PDF
salida_texto =  capture.output({

# Correr modelo JAGS
modelo_binomial = jags(data=datos, inits=iniciales_binomial,
parameters.to.save=cantidades_interes, DIC=F,
n.chains=2, n.iter=55000, n.burnin=5000,
```

```
model.file="BAY - 05.jags") })

# Mostrar resultados del muestreo Gibbs
print(modelo_binomial)
```

```
## Inference for Bugs model at "BAY - 05.jags", fit using jags,
##  2 chains, each with 55000 iterations (first 5000 discarded), n.thin = 50
##  n.sims = 2000 iterations saved
##             mu.vect sd.vect   2.5%    25%    50%    75%  97.5%  Rhat n.eff
## phi           0.264   0.312 -0.332  0.054  0.265  0.467  0.896 1.001  2000
## theta         0.564   0.075  0.418  0.513  0.566  0.615  0.710 1.001  2000
## y_predecida  22.544   4.356 14.000 20.000 23.000 26.000 31.000 1.000  2000
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
```

En la salida del modelo se observan las siguientes salidas para cada valor de interés simulado:

1. **mu.vect:** Es la media (esperanza) de las muestras de la distribución final de las cantidades de interés. Proporciona un estimado central del valor esperado.
2. **sd.vect:** Es la desviación estándar de las muestras de la distribución final de los parámetros Mide la dispersión de la distribución final.
3. **cuantiles:** Son los cuantiles de las muestras de la distribución final.
4. **Rhat:** Es el factor potencial de reducción de escala (PSRF), también conocido como el estadístico de Gelman-Rubin. Se utiliza para evaluar la convergencia de la cadena MCMC. Valores cercanos a 1 indican que las cadenas probablemente han convergido. Por ejemplo, cuando $PSRF = 1.1$ puede ser interpretado como que la distribución final muestrada tiene un potencial de reducción del 10% en el ancho de sus intervalos de probabilidad
5. **n.eff:** Número de muestras efectivas de la distribución final. Indica el número de muestras independientes.