An Example of Bayesian Analysis through the Gibbs Sampler

Hao Zhang

April 16, 2013

1 Gibbs Sampler

The Gibbs sampler is a Monte Carlo method for generating random samples from a multivariate distribution. It is one of the main techniques in Markov chain Monte Carlo. Consider simulating $\boldsymbol{X}$ from a density f. Assume that $\boldsymbol{X}$ can be written as $\boldsymbol{X} = (X_1, \cdots, X_p)$, where the X_i's are either one or multi-dimensional. Moreover, suppose that we can simulate from the full conditionals $f_i(x_i|x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_p)$. The Gibbs sampler proceeds as follows:

`Start with some` $X^{(0)}$`; set` $t = 0$.

`Repeat{`

  `Given` $X^{(t)} = (X_1^{(t)}, \cdots, X_p^{(t)})$,

  `Generate` $X_1^{(t+1)}$ `from` $f_1(\cdot|X_2^{(t)}, \cdots, X_p^{(t)})$

  `Generate` $X_2^{(t+1)}$ `from` $f_2(\cdot|X_1^{(t+1)}, X_3^{(t)}, \cdots, X_p^{(t)})$

  ......

  `Generate` $X_i^{(t+1)}$ `from` $f_i(\cdot|X_1^{(t+1)}, \cdots, X_{i-1}^{(t+1)}, \cdots, X_{i+1}^{(t)} \cdots, X_p^{(t)})$

  `Generate` $X_p^{(t+1)}$ `from` $f_p(\cdot|X_1^{(t+1)}, \cdots, X_{p-1}^{(t+1)})$

  $t = t + 1$;

}

The Gibbs sampler is often used to generate posterior samples from a posterior distribution in a Bayesian framework. The following is an example.

Consider the regression model

$$Y_i = a + bx_i + e_i$$

where e_i are i.i.d $\sim N(0, 1/\tau)$. Assume the prior distributions

$$a \sim N(0, 1/\tau_a)$$
$$b \sim N(0, 1/\tau_b)$$
$$\tau \sim gamma(\alpha, \beta)$$

We will use the Gibbs sampler to generate random samplers from the posterior distribution $f(a, b, \tau|Y_1, \ldots, Y_n)$. It is not hard to derive the following conditional distributions

$$f(a|b, \tau, Y_1, \ldots, Y_n) \sim N\left(\frac{\tau}{n\tau + \tau_a}\sum_{i=1}^{n}(Y_i - bx_i), \frac{1}{n\tau + \tau_a}\right)$$

$$f(b|a,\tau,Y_1,\ldots,Y_n) \sim N\left(\frac{\tau\sum_{i=1}^{n}(Y_i-a)x_i}{\tau\sum_{i=1}(x_i^2+\tau_0)}, \frac{1}{\tau\sum_{i=1}^{n}x_i^2+\tau_b}\right)$$
$$f(\tau|a,b,Y_1,\ldots,Y_n) \sim gamma\left(\alpha+n/2, \beta+(1/2)\sum_{i=1}^{n}(Y_i-a-bx_i)^2\right).$$

The following is a function to implement the Gibbs sampler

```
lm.bayes <- function(y, x, tau.a, tau.b, alpha = 0.001, beta = 0.001, niter = 5000) {
    n <- length(y)
    a <- mean(y)
    b <- 0
    tau <- 1
    result <- matrix(nrow = niter, ncol = 3)
    for (i in 1:niter) {
        a <- rnorm(1, mean = (tau/(n * tau + tau.a)) * sum(y - b * x), sd = 1/sqrt(n * tau +
            tau.a))
        b <- rnorm(1, mean = (tau * sum((y - a) * x))/(tau * sum(x^2) + tau.b), sd = 1/sqrt(tau *
            sum(x^2) + tau.b))
        tau <- rgamma(1, shape = alpha + n/2, rate = beta + 0.5 * sum((y - a - b * x)^2))
        result[i, ] <- c(a, b, tau)
    }
    result
}
```

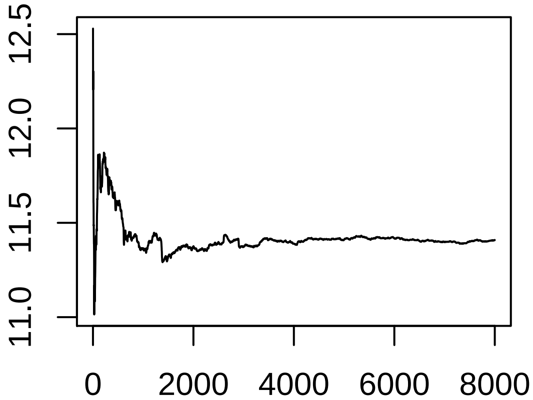
```
data2=data.frame(growth=c(12,10,8,11, 6,7), tannin=0:5)
growth.lm=lm.bayes(y=data2[,1], x=data2[,2], tau.a=0.001, tau.b=0.001, niter=10000)
# Drop the burn-in samples
growth.lm=growth.lm[-(1:2000),]
# posterior means
colSums(growth.lm)/8000

## [1] 11.4086 -0.9648  0.3454
```

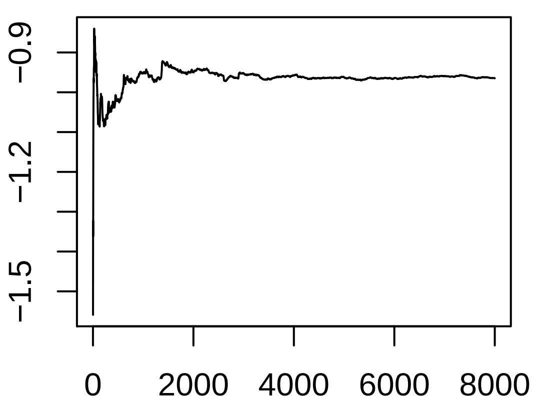
Compare the results with what we have got in class using `jags`. We plot the sample means to check for convergence.

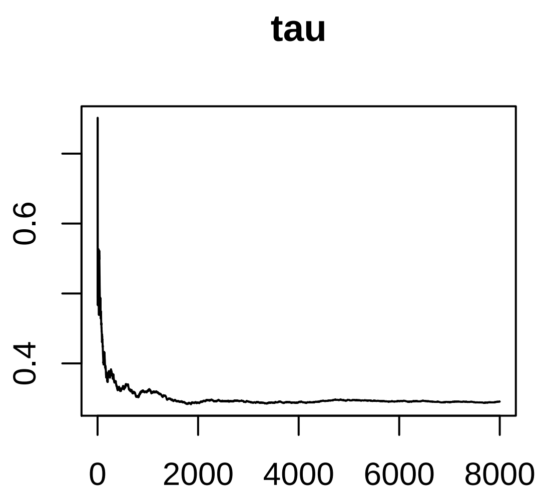
```
plot(cumsum(growth.lm[,1])/(1:8000), type="l", main="a", ylab="", xlab="")
plot(cumsum(growth.lm[,2])/(1:8000), type="l", main="b", ylab="", xlab="")
plot(cumsum(growth.lm[,3])/(1:8000), type="l", main="tau", ylab="", xlab="")
```

a


11.0
11.5
12.0
12.5
0
2000
4000
6000
8000

b


−1.5
−1.2
−0.9
0
2000
4000
6000
8000


tau
0.4
0.6
0
2000
4000
6000
8000