Pontificia Universidad Católica de Chile
Departamento de Estadística
Facultad de Matemática

Profesor: Fernando Quintana
Ayudante: Daniel Acuña León

Ayudantía 10

EYP2805 - Métodos Bayesianos

20 de Octubre

```
## Loading required package: MASS

## Loading required package: rstan

## Loading required package: ggplot2

## Loading required package: StanHeaders

## rstan (Version 2.12.1, packaged: 2016-09-11 13:07:50 UTC, GitRev: 85f7a56811da)

## For execution on a local, multicore CPU with excess RAM we recommend calling
## rstan_options(auto_write = TRUE)
## options(mc.cores = parallel::detectCores())
```

En esta ayudantía indagaremos en como se estima un modelo de regresión lineal simple usando

- Mínimos Cuadrados (`lm()` y definición)
- Gibbs Sampling
- Stan

Sea $Y_i = \beta_1 X_{1i} + \beta_2 X_{2i} + \ldots + \beta_k X_{ki} + \varepsilon_i$, $i = 1, \ldots, n$, con $\varepsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$. La base de datos a utilizar será nuestra vieja amiga *mtcars*.

```r
data(mtcars)
head(mtcars)
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

Quitaremos las covariables categóricas por esta vez, para no complicarnos.

```r
data <- mtcars[, -c(2, 8, 9, 10, 11)]
head(data)
```

```
##                    mpg disp  hp drat    wt  qsec
## Mazda RX4         21.0  160 110 3.90 2.620 16.46
```

```
## Mazda RX4 Wag     21.0  160 110 3.90 2.875 17.02
## Datsun 710        22.8  108  93 3.85 2.320 18.61
## Hornet 4 Drive    21.4  258 110 3.08 3.215 19.44
## Hornet Sportabout 18.7  360 175 3.15 3.440 17.02
## Valiant           18.1  225 105 2.76 3.460 20.22
```

Digamos que queremos predecir los HP versus las demás covariables. Entonces

```r
y <- as.numeric(data[, 3])
x <- as.matrix( cbind(Intercept=1, data[, -3]) )
```

Mínimos Cuadrados

Sabemos que si escribimos este modelo en forma matricial, si se cumplen todos los supuestos y condiciones, el estimador de $\boldsymbol{\beta}$ es

$$\hat{\boldsymbol{\beta}} = (X^t X)^{-1} X^t Y$$

Esto en R es

```r
beta_hat_ls <- solve(t(x) %*% x) %*% t(x) %*% y
beta_hat_ls <- as.numeric(beta_hat_ls)
names(beta_hat_ls) <- colnames(x)
beta_hat_ls
```

```
##   Intercept         mpg        disp        drat          wt        qsec
## 392.8116919  -3.1703151   0.1863757  12.9667790   7.9787211 -16.6811061
```

Que tal los estimadores usando `lm()`?

```r
beta_hat_r <- lm(y ~ . - 1, data=data.frame(x))$coef
beta_hat_r
```

```
##   Intercept         mpg        disp        drat          wt        qsec
## 392.8116919  -3.1703151   0.1863757  12.9667790   7.9787211 -16.6811061
```

Gibbs Sampling

Usandoprioris usuales, $\boldsymbol{\beta} \sim \mathcal{N}_k(\boldsymbol{\beta}_0, \Sigma_0)$, $1/\sigma^2 \sim \text{Gamma}(\nu_0/2, \nu_0\sigma_0^2/2)$, las posterioris completas para los parámetros del modelo de regresión son:

$$\boldsymbol{\beta} \mid Y, X, \sigma^2 \sim \mathcal{N}_k(\boldsymbol{\beta}_n, \Sigma_n)$$

con

$$\Sigma_n = (\Sigma_0^{-1} + X^t X / \sigma^2)^{-1}$$
$$\boldsymbol{\beta}_n = \Sigma_n (\Sigma_0^{-1} \boldsymbol{\beta}_0 + X^t Y / \sigma^2)$$

y

$$\sigma^2 \mid Y, X, \boldsymbol{\beta} \sim \text{Inv-Gamma}\left(\frac{\nu_0 + n}{2}, \frac{\nu_0 \sigma_0^2 + ||Y - X\boldsymbol{\beta}||^2}{2} \right)$$

Con esto, nuestro algoritmo (en pseudo-código) sería así:

1. Actualizar $\boldsymbol{\beta}$:
    a) Calcular $\Sigma_n^{(s)}$ y $\boldsymbol{\beta}_n^{(s)}$
    b) Tomar una muestra de $\boldsymbol{\beta}^{(s+1)} \sim \mathcal{N}_k(\boldsymbol{\beta}_n, \Sigma_n)$
2. Actualizar σ^2:
    a) Calcular $||Y - X\boldsymbol{\beta}^{(s+1)}||^2$
    b) Tomar muestra de $\sigma^{2(s+1)} \sim \text{Inv-Gamma}\left(\frac{\nu_0 + n}{2}, \frac{\nu_0 \sigma_0^2 + ||Y - X\boldsymbol{\beta}^{(s+1)}||^2}{2} \right)$

Y en R esto es fácil. Sigamos de manera modular el procedimiento. Primero, crear una función para tomar muestras de $\boldsymbol{\beta}$.

```
beta_ <- function(beta_0, Sigma_0, X, Y){
    function(sigma2_s){
        Sigma_n <- solve(solve(Sigma_0) + t(X) %*% X / sigma2_s)
        beta_n <- Sigma_n %*% (solve(Sigma_0) %*% beta_0 + t(X) %*% Y / sigma2_s)

        mvrnorm(n = 1, beta_n, Sigma_n)
    }
}
```

Ahora, una para σ^2:

```
sigma2_ <- function(nu_0, sigma2_0, n, X, Y){
    function(beta_s){
        res <- Y - X %*% beta_s
        SSR <- t(res) %*% res

        a <- (nu_0 + n)/2
        b <- (nu_0 * sigma2_0 + SSR) / 2

        1/rgamma(n = 1, a, 1/b)
    }
}
```

Escojamos valores para los hyperparámetros.

```r
k <- ncol(x)
n <- nrow(x)

nu_0 <- 100
sigma2_0 <- 100

beta_0 <- numeric(k)
Sigma_0 <- diag(k) * 100
```

E inicializamos los "muestreadores"

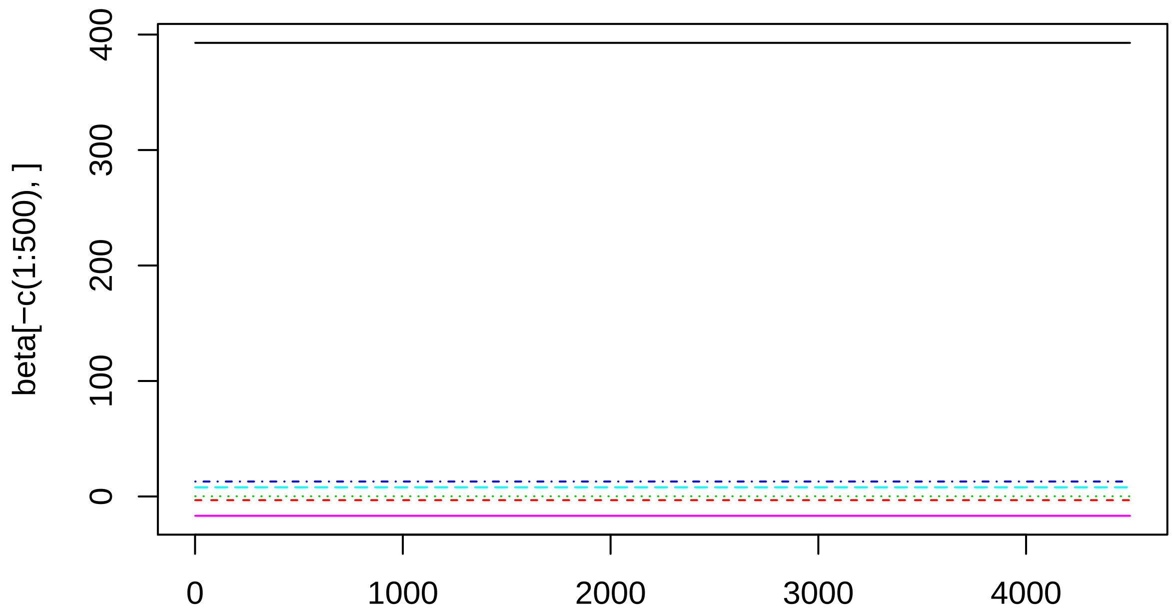
```r
beta_sample <- beta_(beta_0, Sigma_0, x, y)
sigma2_sample <- sigma2_(nu_0, sigma2_0, n, x, y)
```

Ya con esto creado, podemos proceder a hacer un loop desde donde tomaremos las muestras

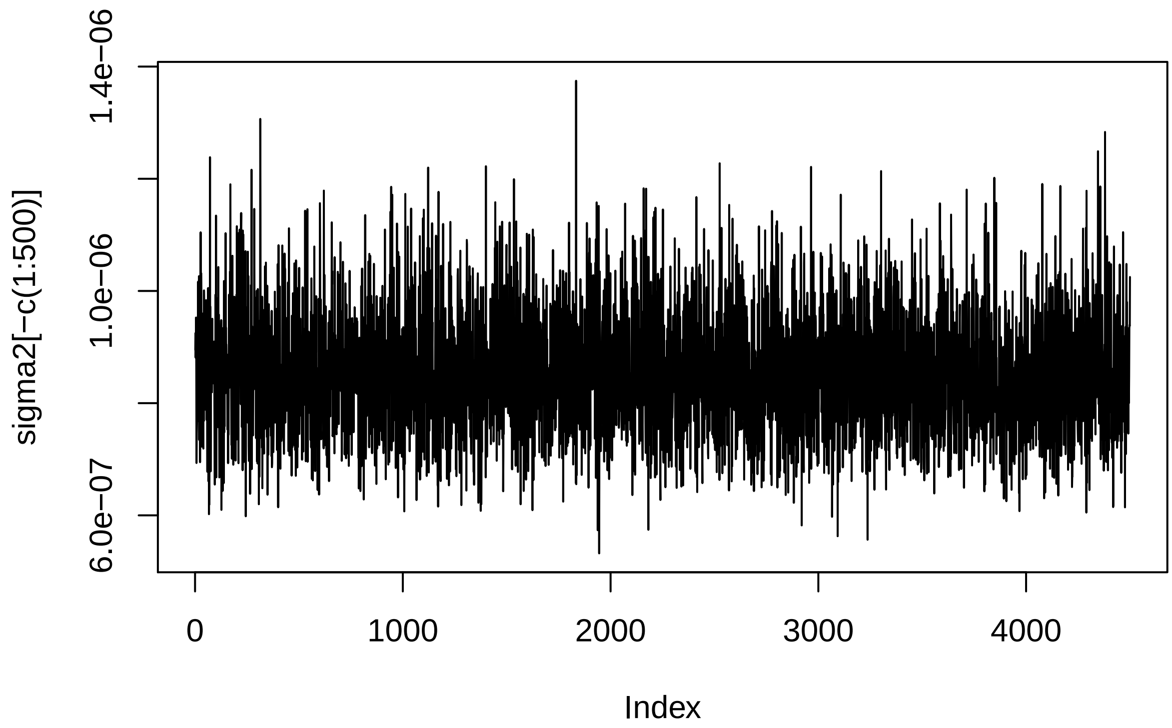
```r
N <- 5000
beta <- matrix(NA, ncol=k, nrow=N)
sigma2 <- numeric(N)*NA

beta[1, ] <- numeric(k) + 1
sigma2[1] <- 100

for(i in 2:N){
    beta[i, ] <- beta_sample(sigma2[i-1])
    sigma2[i] <- sigma2_sample(beta[i, ])
}
```

Analicemos la convergencia de las cadenas:

```r
matplot(beta[-c(1:500), ], type="l")
```




```r
plot(sigma2[-c(1:500)], type="l")
```



Al parecer todas las cadenas exploran con bastante facilidad sus respectivos espacios, con buen *mixing*. Finalmente, los valores estimados son

```
beta_hat_gibbs <- colMeans(beta)
names(beta_hat_gibbs) <- colnames(x)
beta_hat_gibbs
```

```
##   Intercept         mpg        disp        drat          wt        qsec
## 392.6593709  -3.1691867   0.1865919  12.9707443   7.9768016 -16.6756426
```

Stan

Finalmente, volvemos a *Stan*. Primero creamos el modelo.

```
raw_model <- "

data {
  int n;
  int k;
  real y[n];
  matrix[n,k] x;
}
```

```
parameters {
  vector[k] beta;
  real<lower=0> lambda;
}

transformed parameters {
  real<lower=0> sigma;
  vector[n] linpred;

  sigma = inv_sqrt(lambda);
  linpred = x*beta;
}

model {
  for(i in 1:k)
   beta[i] ~ normal(0, 1000);

  lambda ~ gamma(0.001, 0.001);

  y ~ normal(linpred, sigma);
}

"

model <- stan_model(model_code=raw_model)
```

Luego lo ajustamos, i.e. sacamos muestras de la distribución a posteriori.

```
fit <- sampling(model, data=c("n", "k", "y", "x"), iter=5000)
```

Finalmente, tomamos el promedio de esas muestras como nuestro valor estimado.

```
beta_hat_stan <- colMeans(extract(fit, pars="beta")$beta)
names(beta_hat_stan) <- colnames(x)
beta_hat_stan
```

```
##  Intercept        mpg       disp       drat         wt       qsec
## 390.383764  -3.224649   0.188148  13.243676   7.725453 -16.514408
```